

CLAIMS

What is claimed is:

1. A method for enhancing computer application performance comprising:
receiving an application launch argument list;
5 identifying one or more input argument files in the application launch argument list;
creating two or more parallel threads when there are two or more input argument files;
and
processing the input argument files using the parallel threads.
2. The method of Claim 1 wherein creating two or more parallel threads comprises:
10 determining the maximum quantity of parallel threads that can be created; and
creating a quantity of parallel threads according to the quantity of parallel threads that
can be created.
3. The method of Claim 2 wherein determining the quantity of parallel threads that can be
created comprises determining the quantity of parallel threads that can be created
15 according to at least one of a number of active processors, a per-user maximum parallel
thread system limitation and a user-controlled maximum-parallel thread environment
variable.
4. The method of Claim 2 wherein determining the quantity of parallel threads that can be
created comprises determining the quantity of parallel threads that can be created
20 according to one or more environment variables.
5. The method of Claim 2 wherein determining the quantity of parallel threads that can be
created comprises:
receiving a maximum thread indicator from the application launch argument list; and
setting the quantity of maximum parallel threads according to the maximum thread
25 indicator.

6. The method of Claim 1 wherein creating two or more parallel threads comprises:
determining the quantity of input argument files; and
creating a quantity of parallel threads according to the quantity of input argument files.
- 5 7. The method of Claim 1 wherein processing the input argument files comprises:
allocating an input file to a parallel thread;
collecting output from the parallel thread; and
organizing the output from the parallel thread according to the order of input file arguments included in the application launch argument list.
- 10 8. A method for enhancing computer application performance comprising:
receiving an application launch directive;
determining if the application launch directive specifies an application that is a candidate for enhancement;
receiving an application launch argument list;
15 launching two or more parallel instances of the application when the application is a candidate for enhancement and when there are a plurality of input argument files included in the application launch argument list; and
directing to each application instance an instantiation application launch argument list that includes a corresponding one of the input argument files included in the
20 application launch argument list.
9. The method of Claim 8 wherein determining if the application is a candidate for enhancement comprises determining if the application is included in an enumeration of one or more candidate applications.
10. The method of Claim 8 wherein launching two or more parallel instances of the
25 application comprises:
determining the maximum quantity of parallel threads that can be created; and
launching a quantity of parallel instances of the application according to the maximum quantity of parallel threads.

11. The method of Claim 10 wherein determining the maximum quantity of parallel threads that can be created comprises determining the quantity of parallel threads that can be created according to at least one of a number of active processors, a per-user maximum parallel thread system limitation and a user-controlled maximum-parallel thread environment variable.
12. The method of Claim 10 wherein determining the quantity of parallel threads that can be created comprises determining the quantity of parallel threads that can be created according to one or more environment variables
13. The method of Claim 10 wherein determining the quantity of parallel threads that can be created comprises:
receiving a maximum thread indicator from the application launch argument list; and
setting the quantity of maximum parallel threads according to the maximum thread indicator.
14. The method of Claim 8 wherein launching two or more parallel instances of the application comprises:
determining the quantity of input argument files included in the application launch argument list; and
launching a quantity of parallel instances of the application according to the quantity of input argument files.
15. A file processing system comprising:
processor capable of executing an instruction sequence;
memory;
console capable of receiving an argument list;
computer readable medium capable of storing one or more input files and further capable of storing an output stream;
instruction sequence modules stored in the memory including:

argument parser module that, when executed by the processor, minimally causes the processor to identify one or more input argument files in an argument list received by the console;

functional core module that, when executed by the processor, minimally causes the processor to direct an output stream to the computer readable medium according to an input file stored on the computer readable medium; and

task master module that, when executed by the processor, minimally causes the processor to:

create one or more instantiations of the functional core module;
direct to a corresponding instantiation of the functional core module an input argument file identified by the processor when it executes the argument parser; and
cause an assignee processor to execute each instantiation of the functional core module.

16. The file processing system of Claim 15 wherein the task master module minimally causes the processor to create one or more instantiations of the functional core module by minimally causing the processor to:

determine the quantity of parallel threads that can be created; and
create a quantity of instances of the functional core module according to the determined quantity of parallel threads.

17. The file processing system of Claim 16 wherein the task master module minimally causes the processor to determine the quantity of parallel threads that can be created by minimally causing the processor to determine the quantity of parallel threads according to at least one of a number of active processors in a system, a per-user maximum parallel thread system limitation and a user-controlled maximum-thread environment state variable.

18. The file processing system of Claim 16 wherein the task master module minimally causes the processor to determine the quantity of parallel threads that can be created by minimally causing the processor to determine the quantity of parallel threads according to one or more environment variables.
- 5 19. The file processing system of Claim 16 wherein the argument parser, when executed by the processor, further minimally causes the processor to extract a maximum thread indicator from the received argument list and wherein the task master module minimally causes the processor to determine the quantity of parallel threads that can be created by minimally causing the processor to set the quantity of parallel threads according to the
10 maximum thread indicator.
20. The file processing system of Claim 15 wherein the task master module minimally causes the processor to create one or more instantiations of the functional core module by minimally causing the processor to create a quantity of instantiations of the functional core according to the quantity of input argument files included in a received argument
15 list.
21. The file processing system of Claim 15 further comprising an output organizer module that, when executed by a processor, minimally causes the processor to collect output files from one or more instances of the functional core module once they are executed by a processor and further minimally causes the processor to organize the output files
20 according to an order derived by the processor as it executes the argument parser in order to identify input argument files.
22. A file processing system comprising:
processor capable of executing an instruction sequence;
memory;
25 console capable of receiving an application launch directive that includes an argument list;
computer readable medium capable of storing one or more input files and further capable of storing an output stream;

instruction sequence modules stored in the memory including:

command line parser module that, when executed by the processor, minimally causes the processor to:

identify in a received launch directive an application to be executed;
determine if the identified application is a candidate for enhancement;
identify one or more input argument files in an argument list included
in the application launch directive;

generate for a task executive a plurality of load directives and
corresponding instantiation argument lists when the identified
application is a candidate for enhancement and when there are two
or more input argument files in the argument list wherein a
corresponding instantiation argument list includes one of the input
argument files; and

task executive module that, when executed by the processor, minimally causes
the processor to:

load into the memory according to the plurality of load directives and
corresponding instantiation argument lists an application module
that, when executed by the processor, minimally causes the
processor to direct an output stream to the computer readable
medium according to an input file stored on the computer readable
medium;

direct to the application module a corresponding instantiation argument
list generated by the command line parser; and
cause an assignee processor to execute the application module.

23. The file processing system of Claim 22 wherein the command parser module causes the processor to determine if an identified application is a candidate for enhancement by minimally causing the processor to determine if the identified application is included in a pre-established enumeration of candidate applications.

24. The file processing system of Claim 22 wherein the command parser module causes the processor to generate a plurality of load directives and corresponding instantiation argument lists by minimally causing the processor to:

determine the maximum quantity of parallel threads that can be created; and

generate a quantity of load directives and corresponding instantiation argument lists according to the determined maximum quantity of parallel threads.

25. The file processing system of Claim 24 wherein the command parser module minimally causes the processor to determine the maximum quantity of parallel threads that can be created by minimally causing the processor to determine the quantity of parallel threads according to at least one of a number of active processors in a system, a per-user maximum parallel thread system limitation and a user-controlled maximum-thread environment state variable.

26. The file processing system of Claim 24 wherein the command parser module minimally causes the processor to determine the maximum quantity of parallel threads that can be created by minimally causing the processor to determine the quantity of parallel threads according to one or more environment variables.

27. The file processing system of Claim 24 wherein the command line parser, when executed by the processor, further minimally causes the processor to extract a maximum thread indicator from an argument list and wherein the command line parser module minimally causes the processor to determine the quantity of parallel threads that can be created by minimally causing the processor to set the quantity of parallel threads according to the maximum thread indicator.

28. The file processing system of Claim 22 wherein the command line parser causes the processor to generate a plurality of load directives and corresponding instantiation argument lists by minimally causing the processor to:

determine the quantity of input file arguments included in the argument list; and

generate a quantity of load directives and corresponding instantiation argument lists according to the determined quantity of input files.

29. A computer readable medium having imparted thereon instruction sequence modules including:

argument parser module that, when executed by a processor, minimally causes a processor to identify one or more input argument files in a received argument list ;

functional core module that, when executed by a processor, minimally causes a processor to direct an output stream to a computer readable medium according to an input file stored on the computer readable medium; and

task master module that, when executed by a processor, minimally causes a processor to:

create one or more instantiations of the functional core module;

direct to a corresponding instantiation of the functional core module an input argument file identified by a processor when it executes the argument parser; and

cause an assignee processor to execute each instantiation of the functional core module.

30. The computer readable medium of Claim 29 wherein the task master module minimally causes a processor to create one or more instantiations of the functional core module by minimally causing the processor to:

determine the quantity of parallel threads that can be created; and

create a quantity of instances of the functional core module according to the determined quantity of parallel threads.

31. The computer readable medium of Claim 30 wherein the task master module minimally causes a processor to determine the quantity of parallel threads that can be created by minimally causing a processor to determine the quantity of parallel threads according to at least one of a number of active processors in a system, a per-user maximum parallel thread system limitation and a user-controlled maximum-thread environment state variable.

32. The computer readable medium of Claim 30 wherein the task master module minimally causes a processor to determine the quantity of parallel threads that can be created by minimally causing a processor to determine the quantity of parallel threads according to one or more environment variables.
- 5 33. The computer readable medium of Claim 30 wherein the argument parser module, when executed by a processor, further minimally causes a processor to extract a maximum thread indicator from a received argument list and wherein the task master module minimally causes a processor to determine the quantity of parallel threads that can be created by minimally causing a processor to set the quantity of parallel threads according to the maximum thread indicator.
- 10 34. The computer readable medium of Claim 29 wherein the task master module minimally causes a processor to create one or more instantiations of the functional core module by minimally causing a processor to create a quantity of instantiations of the functional core according to a quantity of input argument files included in a received argument list.
- 15 35. The computer readable medium of Claim 15 further comprising an output organizer module that, when executed by a processor, minimally causes a processor to collect output files from one or more instances of the functional core module once they are executed by a processor and further minimally causes a processor to organize the output files according to an order derived by a processor as it executes the argument parser in order to identify input argument files.
- 20 36. A computer readable medium having imparted thereon instruction sequence modules including:
command line parser module that, when executed by a processor, minimally causes a processor to:
25 identify in a received launch directive an application to be executed;
determine if the identified application is a candidate for enhancement;
identify one or more input argument files in an argument list included in the application launch directive;

generate for a task executive a plurality of load directives and
corresponding instantiation argument lists when the identified
application is a candidate for enhancement and when there are two
or more input argument files in the argument list wherein a
5 corresponding instantiation argument list includes one of the input
argument files; and
task executive module that, when executed by a processor, minimally causes a
processor to:

load into the memory according to the plurality of load directives and
10 corresponding instantiation argument lists an application module
that, when executed by a processor, minimally causes a processor
to direct an output stream to a computer readable medium
according to an input file stored on the computer readable medium;
direct to the application module a corresponding instantiation argument
15 list generated by the command line parser; and
cause an assignee processor to execute the application module.

37. The computer readable medium of Claim 36 wherein the command parser module causes
a processor to determine if an identified application is a candidate for enhancement by
minimally causing a processor to determine if the identified application is included in a
20 pre-established enumeration of candidate applications.

38. The computer readable medium of Claim 36 wherein the command parser module causes
a processor to generate a plurality of load directives and corresponding instantiation
argument lists by minimally causing a processor to:
determine the maximum quantity of parallel threads that can be created; and
25 generate a quantity of load directives and corresponding instantiation argument lists
according to the determined maximum quantity of parallel threads.

39. The computer readable medium of Claim 38 wherein the command parser module
minimally causes a processor to determine the maximum quantity of parallel threads that

can be created by minimally causing a processor to determine the quantity of parallel threads according to at least one of a number of active processors in a system, a per-user maximum parallel thread system limitation and a user-controlled maximum-thread environment state variable.

- 5 40. The computer readable medium of Claim 38 wherein the command parser module minimally causes a processor to determine the maximum quantity of parallel threads that can be created by minimally causing a processor to determine the quantity of parallel threads according to one or more environment variables.
- 10 41. The computer readable medium of Claim 38 wherein the command line parser, when executed by the processor, further minimally causes a processor to extract a maximum thread indicator from an argument list and wherein the command line parser module minimally causes a processor to determine the quantity of parallel threads that can be created by minimally causing a processor to set the quantity of parallel threads according to the maximum thread indicator.
- 15 42. The computer readable medium of Claim 36 wherein the command line parser causes the processor to generate a plurality of load directives and corresponding instantiation argument lists by minimally causing the processor to:
- 20 determine the quantity of input file arguments included in the argument list; and
 generate a quantity of load directives and corresponding instantiation argument lists according to the determined quantity of input files.
- 25 43. A file processing system comprising:
 means for receiving an application launch argument list;
 means for identifying one or more input argument files in the application launch argument list;
 means for creating two or more parallel threads when there are two or more input argument files; and
 means for processing the input argument files using the parallel threads.

44. The file processing system of Claim 43 wherein the means for creating two or more parallel threads comprises:

means for determining the maximum quantity of parallel threads that can be created;
and

5 means for creating a quantity of parallel threads according to the quantity of parallel threads that can be created.

45. The file processing system of Claim 44 wherein the means for determining the quantity of parallel threads that can be created comprises a means for determining the quantity of parallel threads that can be created according to at least one of a number of active
10 processors, a per-user maximum parallel thread system limitation and a user-controlled maximum-parallel thread environment variable.

46. The file processing system of Claim 44 wherein the means for determining the quantity of parallel threads that can be created comprises a means for determining the quantity of parallel threads that can be created according to one or more environment variables.

15 47. The file processing system of Claim 44 wherein the means for determining the quantity of parallel threads that can be created comprises:

means for receiving a maximum thread indicator from the application launch
argument list; and

20 setting the quantity of maximum parallel threads according to the maximum thread indicator.

48. A file processing system comprising:

means for receiving an application launch directive;

means for determining if the application launch directive specifies an application that
is a candidate for enhancement;

25 means for receiving an application launch argument list;

means for launching two or more parallel instances of the application when the
application is a candidate for enhancement and when there are a plurality of input
argument files included in the application launch argument list; and

means for directing to each application instance an instantiation application launch argument list that includes a corresponding one of the input argument files included in the application launch argument list.

49. The file processing system of Claim 48 wherein the means for determining if the application is a candidate for enhancement comprises a means for determining if the application is included in an enumeration of one or more candidate applications.

50. The file processing system of Claim 48 wherein the means for launching two or more parallel instances of the application comprises:

means for determining the maximum quantity of parallel threads that can be created;

and

means for launching a quantity of parallel instances of the application according to the maximum quantity of parallel threads.

51. The file processing system of Claim 50 wherein the means for determining the maximum quantity of parallel threads that can be created comprises a means for determining the quantity of parallel threads that can be created according to at least one of a number of active processors, a per-user maximum parallel thread system limitation and a user-controlled maximum-parallel thread environment variable.

52. The file processing system of Claim 50 wherein the means for determining the quantity of parallel threads that can be created comprises a means for determining the quantity of parallel threads that can be created according to one or more environment variables

53. The file processing system of Claim 50 wherein the means for determining the quantity of parallel threads that can be created comprises:

means for receiving a maximum thread indicator from the application launch argument list; and

means for setting the quantity of maximum parallel threads according to the maximum thread indicator.

54. The file processing system of Claim 48 wherein launching two or more parallel instances of the application comprises:

determining the quantity of input argument files included in the application launch argument list; and

5 launching a quantity of parallel instances of the application according to the quantity of input argument files.